Amendments to the Claims:

This listing of claims replaces all prior versions and listings of claims in the application:

Listing of Claims:

23. (Currently amended) A network router comprising:

an input switch;

an output switch;

a controller, the controller comprising a plurality of look-up engines, each of the look-up engines receiving look-up requests in a round robin fashion; and

~~A~~ a memory for storing data for access by a longest prefix match program ~~being executed on a data processing system~~, the program comprising:

a data structure stored in the memory, the data structure including information resident in a database used by the longest prefix match program and including:

a large table at a root, the root branching to nodes containing small trie tables, each trie table addressed by a span of Internet protocol (IP) address bits to locate an indexed trie entry, the indexed trie entry including a route pointer and a trie pointer.

24. (Currently amended) The router ~~memory~~ of claim 23 wherein each of the small trie tables comprises:

prefix match fields for each indexed entry;

a population count of pointers; and

hidden prefix entries.

26. (Currently amended) The router ~~memory~~ of claim 24 wherein the each of the hidden prefix entries hold shorter prefix entry pointers.

27. (Currently amended) The router ~~memory~~ of claim 24 wherein the small trie tables are stored in a static random access memory (SRAM) and used for route lookups, route adds and route deletes.

28. (Currently amended) The router ~~memory~~ of claim 24 wherein the indexed trie entry is a 32-bit longword.

29. (Currently amended) A network router comprising:

a plurality of input ports linked to an input switch;

an output switch linked to a plurality of output ports;

a controller, the controller comprising a plurality of look-up engines, each of the look-up engines receiving look-up requests in a round robin fashion; and

a memory, the memory including a ~~A~~ method of searching a database for a prefix representing a destination address, the method comprising:

reading a data structure stored in ~~a~~ the memory, the data structure comprising a large table at a root, the root branching to two nodes containing small trie tables, each trie table addressed by a span of Internet protocol (IP) address bits to locate an indexed trie entry, the indexed trie entry including a route pointer and a trie pointer;

traversing in parallel the two trie tables of trees to find a match of an trie entry to the prefix.

30. (Currently amended) The router ~~method~~ of claim 29 wherein the route pointer represents the destination address and the trie pointer points to a next small trie table.

31. (Currently amended) The router ~~method~~ of claim 29 wherein the small trie tables comprise:

prefix match fields for indexed table entries;

a population count of pointers; and

hidden prefix entries that hold shorter prefix route entry pointers.

32. (Currently amended) The router ~~method~~ of claim 29 further comprising reporting a non-match if the prefix does not match an entry.

33. (Currently amended) The router ~~method~~ of claim 29 wherein a first large table is a single 64k entry table that is indexed by bits 31:16 of an internet protocol (IP) address.

34. (Currently amended) The router ~~method~~ of claim 29 wherein a second large table is a single 256 entry table that is indexed by bits 31:24 of an internet protocol (IP) address.

35. (Currently amended) The router ~~method~~ of claim 33 wherein the small tables are dynamically allocated and comprise:

a tree with each node representing 4 bits of addresses covering an extension of 1-4 bits of a prefix entry from a previous tree.

36. (Currently amended) The router ~~method~~ of claim 34 wherein the small tables are dynamically allocated and comprise:

a tree with each node representing 4 bits of addresses covering an extension of 1-4 bits of a prefix entry from a previous tree.